Remarks on the Cellular Automaton Global Synchronisation Problem

Nazim Fatès

Remarks on the cellular automaton global synchronisation problem

Nazim Fatès

Inria Nancy Grand-Est, LORIA UMR 7503
F-54 600, Villers-lès-Nancy, France
nazim.fates@inria.fr

Abstract. The global synchronisation problem consists in making a cellular automaton converge to a homogeneous blinking state from any initial condition. We here study this inverse problem for one-dimensional binary systems with periodic boundary conditions (i.e., rings). For small neighbourhoods, we present results obtained with the formulation of the problem as a SAT problem and the use of SAT solvers. Our observations suggest that it is not possible to solve this problem perfectly with deterministic systems. In contrast, the problem can easily be solved with stochastic rules.

Keywords: inverse problems, SAT solving, stochastic vs. deterministic solutions

1 Introduction

The study of inverse problems is becoming a fertile field in the research on cellular automata (CA). Among the recent achievements, we mention the construction of an exact solution to the parity problem [**?**], the construction of exact or approached solutions on the density classification problem [**?**,**?**,**?**,**?**] or the reinterpretation of the solution to the Firing squad problem with fields [**?**].

A common feature of these problems is the need to reach a global consensus: there exists a moment where all cells, or a large fraction, must agree on a given state that is interpreted as the output of the algorithm. The difficulty is related to the propagation of this information from a local to a global scale: in a decentralised framework such as cellular automata, how do the cells "agree" on a common state while they have only a local view of the system?

We here study the *global synchronisation problem*: in its original form, the question is to find a CA rule such that, from any initial condition, the system reaches a "blinking state" in which the two homogeneous configurations alternate. This problem can be generalised to more states but we will here restrict our study to the binary case.

Since its formulation by Das et al. in 1994 [**?**], the problem has received only a limited attention. This lack of interest is probably due to the fact that it is much easier to solve than other inverse problems such as the density classification problem. In fact, solutions with "100% success rates" were presented in the

very paper where the problem was formulated [**?**]. The authors used genetically engineered solutions to show that it was possible to obtain a performance of "100%" for ring sizes going up to 999. Their interest was to find rules which attain a consensus by removing the "defects" that separate the non-synchronised regions of the system.

Our purpose in this note is to go one step forward and to ask if perfect solutions do exist. We will thus request that *all* initial conditions lead to the blinking cycle and not only a sample of configurations, drawn at random. After presenting the formal definitions of the problem (Sec. **??**), we will present a simple "manual proof" that no ECA solves the problem (Sec. **??**). This construction will guide us to formulate the problem as a SAT problem (Sec. **??**) and to obtain a first set of results for larger neighbourhoods (Sec. **??**). We then show that, in contrast, perfect solutions can easily been constructed (Sec. **??**). We conclude by formulating a few questions.

2 Fundamentals

2.1 CA definitions

We here consider *finite* binary cellular automata with periodic boundary conditions. The basic components of our systems are the *cells*; each cell can hold one of the two states: $\mathtt{0}$ or $\mathtt{1}$. The variable n is used to denote the number of cells that compose the system, the cells are arranged in a ring and the set of cells is denoted by $\mathcal{L} = \mathbb{Z}/n\mathbb{Z}$.

A *configuration* $x = (x_i)_{i\in\mathcal{L}}$ represents the state of the system at a given time. The set of configurations is denoted by $\mathcal{E}_n = \{\mathtt{0},\mathtt{1}\}^{\mathcal{L}}$. The interactions between the cells are local, that is, each cell can only "see" a finite subset of the cells of the system, the *neighbourhood*. Without loss of generality, we can consider that the neighbourhood $\mathcal{N}$ of our cellular automata are formed of discrete intervals: $\mathcal{N} = \{-l,\dots,r\}$, with $l \in \mathbb{N}$ and $r \in \mathbb{N}^* = \{1,2,\dots\}$. The width of this interval is called the *size* of the neighbourhood and is denoted by k, with $k = l + r + 1$.

The evolution of a cell follows a function $f : \{\mathtt{0},\mathtt{1}\}^k \to \{\mathtt{0},\mathtt{1}\}$, called the *local rule*. Following Wolfram's notation, a local rule of size k, that is, defined on a neighbourhood of size k, is assigned a code W which is an integer between 0 and $2^k - 1$. This code is given by the formula: $W = \sum_{i=0}^{2^k-1} f(\mathfrak{b}_k(i),\dots,\mathfrak{b}_1(i)) \cdot 2^i$ where $\mathfrak{b}_j(i)$ is the value of the j-th bit of the binary representation of i.

For a given ring size n, the global transition function $F : \mathcal{E}_n \to \mathcal{E}_n$ associated to the ring size n is the function that maps a configuration x^t to a configuration $x^{t+1} = F(x^t)$ such that $x^0 = x$ and:

$$\forall i \in \mathcal{L},\ x_i^{t+1} = f(x_{i-l},\dots,x_{i+r}). \tag{1}$$

Note that to be perfectly rigorous, we should denote F with indices showing that it depends on f and n. We however drop these elements for the sake of clarity since f and n will be made clear from the context.

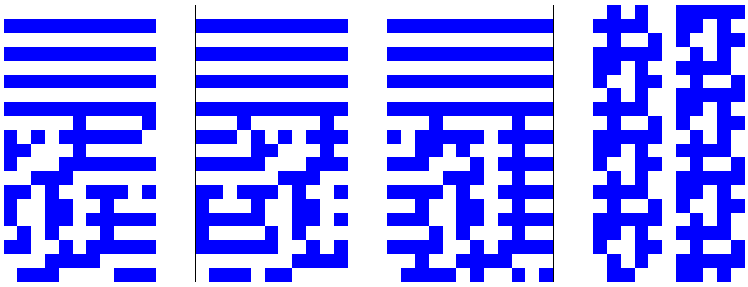

Fig. 1. Four space-time diagrams for rule 1078270911 with $k = 5$ (see page **??**). Time goes from bottom to top, white and blue squares represent cells in state 0 and 1, respectively. (a), (b): synchronised initial conditions with $n = 11$, (c) and (d): initial conditions with $n = 12$, the last one is not synchronised.

2.2 Formulation of the problem

We denote by $\mathbf{0} = 0^{\mathcal{L}}$ and $\mathbf{1} = 1^{\mathcal{L}}$ the two *uniform* configurations. In the following, we will require that 0 and 1 are two non-quiescent states, that is, $f(0, \ldots, 0) = 1$ and $f(1, \ldots, 1) = 0$. We call this condition the *blinking condition*.

We define the *height* h of a configuration $x \in \mathcal{E}_n$ as the time needed to reach one of the two uniform configurations:

$$h(x) = \min\{t \in \mathbb{N}, x^t = F^t(x) \in \{\mathbf{0}, \mathbf{1}\}\}. \tag{2}$$

with the convention that $h(x) = \infty$ if x^t does not reach $\mathbf{0}$ or $\mathbf{1}$. Similarly, the *height* of a given configuration space $\mathcal{E}_n$ is the maximum height of the configurations of $\mathcal{E}_n$.

We say that F *synchronises* a configuration $x \in \mathcal{E}_n$ if $h(x)$ is finite. We also say that x is synchronised on $\mathbf{0}$ (resp. on $\mathbf{1}$) if the first homogeneous configuration that is met is $\mathbf{0}$ (resp. $\mathbf{1}$). Similarly, we say that F synchronises the size n if it synchronises all the configurations of $\mathcal{E}_n$. We can now formulate the global synchronisation problem:

> Does there exist a local rule f such that for all $n \in \mathbb{N}^*$, the associated global function F synchronises the size n?

2.3 Elementary properties

Let f be a local rule of size k. The *reflexion* $R(f)$ and the *conjugate* $C(f)$ are the local rules respectively obtained by the exchange of the left and right directions and by the exchange of the 0 and 1 states. Formally, $\forall (q_1, \ldots, q_k) \in \{0, 1\}^k$,

$$R(f)(q_1, \ldots, q_k) = f(q_k, \ldots, q_1),$$

and

$$C(f)(q_1, \ldots, q_k) = \overline{f(\overline{q_1}, \ldots, \overline{q_k})},$$

where $\overline{q} = 1 - q$ denotes the inversion of states. Similarly, the reflexion-conjugate rule $RC(f)$ is the local rule obtained by composing the two previous symmetries: $RC(f) = R \circ C(f) = C \circ R(f)$.

Proposition 1 (rule symmetries). *f is a solution to the global synchronisation problem if and only if $R(f), C(f), RC(f)$ are also solutions.*

Proof. Clearly, the property of synchronising a given size is preserved by the reflection and conjugation symmetries: for a given size n, if $F, F_{\mathrm{r}}, F_{\mathrm{c}}, F_{\mathrm{rc}}$ are the global functions respectively associated to $f, R(f), C(f)$ and $RC(f)$, then F synchronises the size n if and only if $F_{\mathrm{r}}, F_{\mathrm{c}}, F_{\mathrm{rc}}$ synchronise the size n. □

Let σ denote the (left) shift operator, that is, a function $\sigma : \mathcal{E}_n \to \mathcal{E}_n$ such that $\forall i \in E, \sigma(x)_i = x_{i+1}$. We call the *rotations* of x the set of configurations that are obtained by applying a positive number of shifts on x; this set is denoted by $[x] = \{\sigma^k(x), k \in \mathbb{N}\}$.

Proposition 2 (configuration symmetries). *A global rule F synchronises a configuration x if and only if it synchronises all the configurations of $[x]$.*

This simply results from the fact that: (a) F commutes with the shift and (b) $[\mathbf{0}] = \{\mathbf{0}\}$ and $[\mathbf{1}] = \{\mathbf{1}\}$. Note that the rotation $[\cdot]$ defines an equivalence class: we say that a configuration y is equivalent to x if y is a rotation of x. It can be easily verified that this is an equivalence relation.

The next proposition states that the iterates of a configuration can not be contained in the rotations of this configuration.

Proposition 3 (images of a configuration). *If F synchronises a configuration x, then $\left(\cup_{k\geq 1} F^k([x])\right) \bigcap [x] = \emptyset$, that is, $\forall x' \in [x], \forall k \in \mathbb{N}^*, F^k(x') \notin [x]$.*

Proof. By contradiction, let us assume that there exists $k \in \mathbb{N}^*$ and $i \in \mathbb{N}$ such that $F^k(x) = \sigma^i(x)$. By recurrence, using the commutation of the shift with F, we have: $F^{kn}(x) = \sigma^{in}(x)$. Since the space is a ring of size n, we have $\sigma^{in}(x) = x$, which implies $x = F^{kn}(x)$. The configuration x thus evolves on a cycle of length k. The two homogeneous states $\mathbf{0}$ and $\mathbf{1}$ are excluded from this cycle – otherwise x would be reachable from these two states – and x can not be synchronised, which contradicts the hypothesis. □

An immediate consequence of this proposition is that if F synchronises x, then x can not be a *fixed point* ($F(x) = x$) or a *blinking point* ($F(x) = \overline{x}$ and $F(\overline{x}) = x$) or a *translating point* ($F(x) = \sigma^i x$ with $i \in \mathbb{N}^*$).

Proposition 4 (color-discernation). *If a local rule f is a solution to the problem, then it is not color-blind, that is, $C(f) \neq f$.*

Proof. By contradiction, let us assume that we have $C(f) = f$. Let us take an even size $n = 2m$ with $m \in \mathbb{N}$ and consider the configuration $x = (\mathtt{01})^m$. Without loss of generality we can assume that x is synchronised on $\mathbf{0}$. Formally, if we denote by $T = h(x)$ the height of x, this reads: $F^T(x) = \mathbf{0}$.

Then, if we consider $\overline{x} = (\overline{x_i})_{i \in \mathcal{L}}$, we have: $F_c^T(\overline{x}) = \overline{F^T(x)} = \overline{\mathbf{0}} = \mathbf{1}$.

On the other hand: $F^T(\sigma x) = \sigma F^T(x) = \sigma \mathbf{0} = \mathbf{0}$. Since $\overline{x} = \sigma x$, we remark that these two equations are contradicting. □

3 Direct inspection of the ECA space

As a starting point, let us consider Elementary Cellular Automata (ECA), that is, binary CA with $\mathcal{N} = \{-1, 0, 1\}$.

Proposition 5. *There exists no ECA which solves the synchronisation problem.*

Proof. We have 256 rules to consider, these rule are defined with:
$\mathsf{a} = f(0,0,0)$ $\mathsf{b} = f(0,0,1)$ $\mathsf{c} = f(1,0,0)$ $\mathsf{d} = f(1,0,1)$
$\mathsf{f} = f(0,1,1)$ $\mathsf{e} = f(0,1,0)$ $\mathsf{g} = f(1,1,0)$ $\mathsf{h} = f(1,1,1)$.
We now have to determine whether there exists an assignment of these eight boolean variables $\mathsf{a}, \ldots, \mathsf{h}$ which satisfies the problem.

Case $n = 1$**:** Given the specification of the problem, the states $\mathtt{0}$ and $\mathtt{1}$ are not quiescent: $\mathsf{a} = 1$ and $\mathsf{h} = 0$. We are thus left with 64 rules to search.

Case $n = 2$**:** Proposition **??** implies $\mathsf{d} = \mathsf{e}$. Indeed, if the two variables are not equal, $\mathtt{01}$ would be either a fixed point or a blinking point. We are now left with 32 rules.

Case $n = 3$**:** By noting that the configuration $\mathtt{001}$ can not be a fixed point and can not be translated (by Prop. **??**), we obtain: $s_1 = \mathsf{b} + \mathsf{c} + \mathsf{e} \neq 1$ (CondA). By symmetry, we have $s_2 = \mathsf{d} + \mathsf{f} + \mathsf{g} \neq 2$ (CondB).

The case where a configuration of $[\mathtt{001}]$ is transformed into a configuration of $[\mathtt{011}]$ and vice-versa is also impossible, otherwise the uniform configurations would never be reached (Prop. **??**). We thus have $(s_1, s_2) \neq (2, 1)$ (CondC). It is then easy to verify that these conditions are sufficient to achieve the synchronisation of size 3.

We find the following set of remaining 8 rules, divided into three sets of rules: $(1, 127)$, $(9, 65, 111, 125)$, $(19, 55)$. The rules are grouped by their equivalence with the conjugation and reflexion symmetries (see Prop. **??**).

Case $n = 4$**:** It is easy to check that none of the remaining rules allows a synchronisation of size 4. For instance, we remark that $\mathtt{0001}$ is a translating point for rule 1 and 9 and that $\mathtt{0110}$ is a translating point for rule 19. □

4 The synchronisation problem as a SAT problem

The previous method required a methodical inspection of the space but this approach can not easily been generalised to larger neighbourhoods. We now propose to do an "automated filtering" of the rules by transforming the problem

into a SAT problem. The idea is to model the transition table as sequence of boolean variables and to express a boolean formula to state that a configuration is synchronised. We want to examine more and more initial conditions until we reach a point where we find that the problem is not satisfiable. If, on the contrary, we find that there exists a rule which synchronises all the initial conditions considered, then we would have a good "candidate" for solving the problem.

4.1 General SAT formulation

A SAT problem consists in finding a assignment to boolean variables that satisfies a given boolean formula. A *clause* is a conjunction of literals, that is, a boolean formula defined only with the `or` operator (e.g., $a \vee b \vee c$). The convention is that a SAT problem is formulated in a conjunctive normal form (CNF): it is a disjunction of clauses. In the sequel, we call a *CNF formula* any of such disjunction of clauses.

Let k be the size of the neighbourhood and $M = 2^k$ the number of transitions[1] of this neighbourhood. We introduce M boolean variables $t_0, \ldots, t_{M-1}$ to encode the transitions of a rule f; the convention is that t_i is true if and only if $f\big(\mathfrak{b}_k(i), \ldots, \mathfrak{b}_1(i)\big) = 1$. For an initial condition $x \in \mathcal{E}_n$, we also introduce $(\tau + 1) \cdot n$ additional variables, denoted by $(\xi(t,i))_{t\in\{0,\ldots,\tau\},i\in\{0,\ldots,n-1\}}$, which correspond to the values $(x_i^t) \in \{\mathtt{0},\mathtt{1}\}$ taken by the cells in the evolution of x.

4.2 Blinking condition

The blinking condition $f(q,\ldots,q) = \overline{q}$ is simply expressed by a CNF formula with two atomic clauses: $\mathrm{F}_{\mathrm{bl}} = t_0 \wedge \neg t_{M-1}$.

4.3 Initial state

Let us now see how to encode the states of an initial condition $x \in \mathcal{E}_n$ in a formula. The operation simply consists in "translating" the initial condition x into the CNF formula:

$$\mathrm{F}_{\mathrm{ic}}(x) = \bigwedge_{i\in\{0,\ldots,n-1\}} \mathbb{1}\{\xi(0,i), x_i^0\}. \tag{3}$$

where $\mathbb{1}\{V,q\}$ is a function which associates to the boolean variable V and to a cell state $q \in \{\mathtt{0},\mathtt{1}\}$ the variable V if $q = \mathtt{1}$ and the variable $\neg V$ otherwise.

For instance if we have $x = \mathtt{0011}$ as an initial condition, the associated formula will be:

$$\mathrm{F}_{\mathrm{ic}}(x) = \neg\xi(0,0) \wedge \neg\xi(0,1) \wedge \xi(0,2) \wedge \xi(0,3). \tag{4}$$

[1] A transition is a tuple of size k which is given as an input to the local rule.

4.4 Synchronisation condition

Let τ be the maximum number of time steps to achieve the synchronisation. To express the condition that x is synchronised in at most τ steps, we can write $x^\tau \in \{\mathbf{0}, \mathbf{1}\}$. Unfortunately, this can not be translated in a straightforward way into a CNF formula. Indeed, if we write:

$$\mathrm{F}_{\mathrm{sc}}(x, \tau) = \left(\bigwedge_{i \in \{0,\ldots,n-1\}} \xi(\tau, i) \right) \bigvee \left(\bigwedge_{i \in \{0,\ldots,n-1\}} \neg\xi(\tau, i) \right), \tag{5}$$

we need to distribute the `and` operator over the `or`to obtain a CNF. This is why we prefer to formulate this condition as:

$$\mathrm{F}_{\mathrm{sc}}(x, \tau) = \bigwedge_{i \in \{0,\ldots,n-2\}} \xi(\tau, i) = \xi(\tau, i+1), \tag{6}$$

which simply represents the fact that all the states of x^τ are equal. By noting that $a = b$ is equivalent to $(a \vee \neg b) \wedge (\neg a \vee b)$, F_{sc} becomes:

$$\mathrm{F}_{\mathrm{sc}}(x, \tau) = \bigwedge_{i \in \{0,\ldots,n-2\}} (\xi(\tau, i) \vee \neg\xi(\tau, i+1)) \wedge (\neg\xi(\tau, i) \vee \xi(\tau, i+1)). \tag{7}$$

4.5 Consistency conditions

We now need to write a CNF formula for the condition: $x^{t+1} = F(x^t)$ for $t \in \{0, \ldots, \tau - 1\}$. We call this formulation the "*consistency condition*", as it expresses the fact a given boolean formula in consistent with the evolution of the cellular automaton. We now give a precise description of this CNF formula. In order to ease the notations, let us detail this operation for the specific case of $\mathcal{N} = \{-1, 0, 1\}$; it is easy to generalise it to other neighbourhoods. Locally, our condition is expressed by

$$\forall i \in \mathcal{L},\ x_i^{t+1} = f(x_{i-1}^t, x_i^t, x_{i+1}^t) \tag{8}$$

which is translated as:

$$\forall i \in \{0, \ldots, n-1\}, \varphi(\xi(t+1, i), \xi(t, i^-), \xi(t, i), \xi(t, i^+)) \tag{9}$$

where $i^- = (i-1) \mod n$ and $i^+ = (i+1) \mod n$, and where φ is a function that remains to be found.

Our goal is to find φ such that $\varphi(y', x, y, z)$ is a CNF formula that expresses that y' is the result of transition $f(x, y, z)$. In a usual programming environment, one would need simply to calculate $i = x + 2y + 4z$ and then to read the value of t_i and assign it to y'. Unfortunately, there is no direct way of "coding" these operations in a SAT formula. We thus need to enumerate all the possible values for the variables y', x, y, z and then write a consistency condition that expresses that y' equals t_i where i is the index which corresponds to the transition (x, y, z).

For example, if we take transition $f(0,1,1)=1$, we have $i=3$ and we write the formula with five clauses:

$$\varphi_3 = \neg x \wedge y \wedge z \wedge t_3 \wedge (y' \vee \neg t_3) \wedge (\neg y' \vee t_3), \tag{10}$$

where the two last clauses stand for $y' = t_3$.

Formally, we write: $\varphi(y', x, y, z) = \bigvee_{\lambda \in \{0,\dots,7\}} \varphi_\lambda$ with:

$$\begin{aligned} \varphi_\lambda = {} & \mathbb{1}\{x, \lambda_1\} \wedge \mathbb{1}\{y, \lambda_2\} \wedge \mathbb{1}\{z, \lambda_3\} \wedge \\ & \mathbb{1}\{t_\lambda, f(x,y,z)\} \wedge (y' \vee \neg t_\lambda) \wedge (\neg y' \vee t_\lambda), \end{aligned} \tag{11}$$

where $\lambda_i = \mathfrak{b}_i(\lambda)$ is the value of the i-th bit of the binary representation of λ. By distributing the `and` operator over the `or`, φ_λ becomes:

$$\begin{aligned} \varphi_\lambda = {} & (\mathbb{1}\{x, \overline{\lambda_1}\} \vee \mathbb{1}\{y, \overline{\lambda_2}\} \vee \mathbb{1}\{z, \overline{\lambda_3}\} \vee y' \vee \neg t_\lambda) \bigwedge \\ & (\mathbb{1}\{x, \overline{\lambda_1}\} \vee \mathbb{1}\{y, \overline{\lambda_2}\} \vee \mathbb{1}\{z, \overline{\lambda_3}\} \vee \neg y' \vee t_\lambda). \end{aligned} \tag{12}$$

Each elementary transition of a given cell at a given time step is thus encoded with a formula φ which contains $2M = 2^{k+1} = 16$ clauses. As there are $n\tau$ such elementary conditions, the consistency condition is given by $\mathrm{F_e}$ with $2^{k+1} \cdot n\tau$ clauses:

$$\mathrm{F_e}(x, \tau) = \bigwedge_{\substack{t \in \{0,\dots,\tau-1\}, \\ i \in \{0,\dots,n\}}} \varphi\big(\xi(t+1,i), \xi(t,i^-), \xi(t,i), \xi(t,i^+)\big). \tag{13}$$

4.6 Combining initial conditions

The last step that remains is to combine various initial conditions in order to: (a) either find out that the problem is not solvable for a given setting or (b) exhibit a good candidate to solve the problem.

We proceed iteratively by increasing the size of the initial conditions to synchronise. From Prop. **??**, we know that we do not need to consider all the initial conditions: for each size n, it is sufficient to select only *one* initial condition in each possible set of rotations. Formally, we say that a set of configurations is *representative* if the rotations of its members form a partition of the configuration space. Formally, let us denote by $\chi(n)$ the set of representative conditions; we write:

$$\chi(n) = \{X \subset \mathcal{E}_n, \forall x, y \in X, [x] \cap [y] = \emptyset, \bigcup_{x \in X} [x] = \mathcal{E}_n\}. \tag{14}$$

The following table shows the growth of these sets[2]:

n	1	2	3	4	5	6	7	8	9
$\lvert\chi(n)\rvert$	2	3	4	6	8	14	20	36	60

[2] It corresponds to sequence `A000031` in the *Online Encyclopedia of integer sequences*. One reads: "In music, $|\chi(n)|$ is the number of distinct classes of scales and chords in an n-note equal-tempered tuning system", see: `https://oeis.org/A000031`

To sum up, if we fix a set of ring sizes $\mathcal{S} = \{n_1, \ldots, n_s\}$, and a maximum synchronisation time τ, we construct a sequence of *sets* of configurations: $X_1, \ldots, X_s$ such that $X_i \in \chi(n_i)$ and build the *general formula*:

$$\mathrm{F}_{\mathrm{synch}}(\mathcal{S}, \tau) = \mathrm{F}_{\mathrm{bl}} \bigwedge_{i \in \{1,\ldots,s\}} \bigwedge_{x \in X_i} \mathrm{F}_{\mathrm{ic}}(x, \tau) \wedge \mathrm{F}_{\mathrm{sc}}(x, \tau) \wedge \mathrm{F}_{\mathrm{e}}(x, \tau). \qquad (15)$$

This is the final formula; it expresses the fact that all the configurations of size $n \in \mathcal{S}$ are synchronised in at most τ time steps.

5 First experimental results

The use of SAT solvers is a well-explored field of research in Computer science. As we are not a specialist of these questions, we did not endeavour to optimise the search for a solution by any means. We simply used the `minisat` solver[3] and generated the formulae with our cellular automata simulation program `FiatLux`[4].

5.1 ECA space

We take $k = 3$. The results with the SAT solver confirm the results of Sec. **??**: there exists no rule which is a solution for $\mathcal{S} = \{2, 3, 4\}$. For $\mathcal{S} = \{2, 3\}$, we find that:

- $(1, 127)$ have height of 1,
- $(19, 55)$ and $(9, 65, 111, 125)$ have height of 2, and
- no rule has height of 3.

We can also explore the synchronisation for other sets of ring sizes $\mathcal{S}$. For $\mathcal{S} = \{4\}$, we find that:

- $(37, 91)$ have a height of 3,
- $(25, 67, 103, 61)$ and $(45, 101, 75, 89)$ have height of 4 and,
- no rule has a height of 5.

For ECA 61, the synchronisation process is presented on Fig. **??**, p. **??**. Surprisingly, for $\mathcal{S} = \{5\}$, we find that $(9, 65, 111, 125)$ synchronises with a height of 5.

5.2 The $k = 4$ space

We now examine a neighbourhood with one more cell: we take $k = 4$ and $\mathcal{N} = \{-1, 0, 1, 2\}$. The space contains $2^{2^4} = 2^{16} = 65536$ rules.

By testing increasing values of n and setting the maximum synchronisation time τ equal to $\chi(n)$ (see Prop. **??**), we found that the maximum synchronisation length of this neighbourhood is 6, that is, for $\mathcal{S} = \{2, \ldots, 7\}$, no solution is found.

For $\mathcal{S} = \{2, \ldots, 6\}$, we find that:

[3] see: `http://minisat.se/`

[4] see `http://fiatlux.loria.fr`

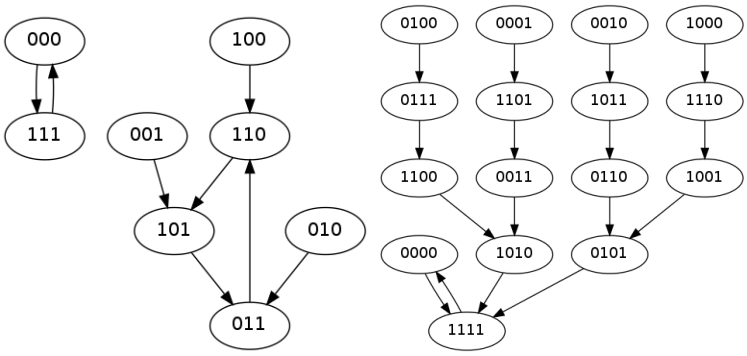

Fig. 2. Transition graphs of ECA 61 for $n = 3$ and $n = 4$. An oriented link between a configuration x and y represents the relationship $y = F(x)$.

- 6 rules have a height of 4: (1077,21471), (4427,11639), (11893,20875),
- 6 rules have a height of 5: (1205,17461,21215,21469), (5419,11095),
- 2 rules have a height of 6: (4363,12151).

For rule 5419, the synchronisation process is presented on Fig. **??**, p. **??**.

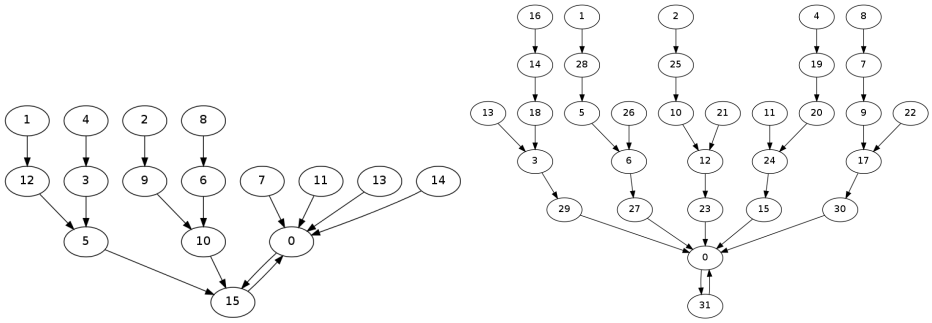

Fig. 3. Neighbourhood of size $k = 4$: transition graphs of rule 5419 for $n = 4$ and $n = 5$. For the sake of readability configurations have been represented by a number which corresponds to the decimal conversion of their bits.

5.3 The $k = 5$ space

We now examine a neighbourhood with one more cell: we take $k = 5$ and $\mathcal{N} = \{-2, -1, 0, 1, 2\}$. The space contains $2^{2^5} = 2^{32} \sim 4.10^{10}$ rules. At this point, we reach the limits of our approach: The CNF formula of the problem has 74768 variables and 4563060 clauses. By progressively increasing the values of n and τ, our best result was to find a rule which synchronises the size interval $\mathcal{S} = \{2, \ldots, 11\}$: rule 1078270911. This rule has a height of 18 (see Fig. **??**, p. **??**).

There are probably other rules which solve the problem for τ higher than 18, but we leave this exploration for future work. For $\mathcal{S} = \{2, \ldots, 12\}$, no solution was found for $\tau = 30$. Surprisingly, the non-satisfiability of the formula is given rapidly, which suggests that the maximum synchronisation length for $k = 5$ is equal to 12.

We also tested the SAT solver for $k = 6$, but $\tau = 12$ is sufficient to generate SAT problems that are not solved after more than two hours of computation. We thus leave the exploration of these greater spaces for future work.

6 "Perfect" stochastic solutions

We now propose to examine what is the situation of the stochastic rules. In fact, if we allow randomness in the transitions of the rule, it becomes difficult *not* to solve the problem! For the sake of simplicity we restrict our study to the probabilistic ECA case. We thus take $\mathcal{N} = \{-1, 0, 1\}$ and define a local transition function $\phi : Q^3 \to [0, 1]$, which associates to each neighbourhood state its probability to be updated to 1.

Formally, starting from a configuration x, the system can be described by a *stochastic process* $(x^t)_{t\in\mathbb{N}}$. The sequence (x^t) now denotes a sequence of *random variables*, which is constructed recursively with: $x^0 = x$ (with probability 1) and

$$\forall t \in \mathbb{N},\ \forall i \in \mathcal{L},\ x_i^{t+1} = \mathcal{B}(\phi(x_{i-1}^t, x_i^t, x_{i+1}^t)), \tag{16}$$

where $\mathcal{B}(p)$ is the Bernoulli random variables, which equals 1 with probability p and 0 with probability $1 - p$. Note that strictly speaking, the definition above is more a characterisation than a definition and that a "proper" definition would require the use of tools from measure theory (see e.g. [**?**]).

We now need to redefine what it means to solve the problem perfectly. The blinking condition is easily translated to $\phi(0, 0, 0) = 1$ and $\phi(1, 1, 1) = 0$. For a rule which verifies the blinking condition and a given configuration $x \in \mathcal{E}_n$, we define the synchronisation time $T(x)$ as the random variable which corresponds to the number of steps needed to attain one of the two homogeneous configurations: $T(x) = \min\{t, x^t \in \{\mathbf{0}, \mathbf{1}\}\}$. The average synchronisation time of x is the expectancy of $T(x)$, denoted by $\mathbb{E}\{T(x)\}$. For a given size n, we define the worst expected synchronisation time (WEST) of x and the expected average synchronisation time (EAST) of x as:

$$\mathrm{WEST}(n) = \max_{x\in\mathcal{E}_n} \mathbb{E}\{T(x)\}, \tag{17}$$

$$\mathrm{EAST}(n) = \tfrac{1}{2^n} \sum_{x\in\mathcal{E}_n} \mathbb{E}\{T(x)\}. \tag{18}$$

We say that f *synchronises* the size n if WEST(n) is finite. Clearly, this is equivalent as having a finite EAST(n). By extension, f is a solution to the global synchronisation problem if f synchronises all sizes $n \in \mathbb{N}$.

Let us now examine how to build a solution. For a function ϕ, we introduce the variables: $p_0 = \phi(0,0,0)$, $p_1 = \phi(0,0,1)$, $\ldots$, $p_7 = \phi(1,1,1)$.

Proposition 6. *Let ϕ be a probabilistic ECA such that $p_0 = 1$, $p_7 = 0$ and $\forall i \in \{1,\ldots,6\}, p_i \in]0,1[$, then ϕ is a solution to the global synchronisation problem.*

Proof (Sketch). To show that $T(x)$ is finite for every $x \in E$, it is sufficient to show that there is a non-zero probability to reach $\mathbf{0}$ from x in a finite number of steps. Let $Z(x)$ be the function such that $Z(x)_i = \begin{cases} 1 & \text{if } (x_{i-1}, x_i, x_{i+1}) = (0,0,0), \\ 0 & \text{otherwise .} \end{cases}$
The effect of Z is to change a cell to a 0 whenever there is a non-zero probability that this cells updates to 0. Let us denote by m the greatest time needed to shrink the regions of ones; clearly, $m = \lfloor n/2 \rfloor$. It can then easily be checked that:

(a) $\forall x \neq \mathbf{1}, Z^m(x) \in \{\mathbf{0}, \mathbf{1}\}$, and

(b) there is a non-zero probability to have $\forall t \in \{1,\ldots,m\}, x^t = Z^t(x)$.

In other words, there is a non-zero probability that x is absorbed in the **0**-**1** cycle in at most m time steps. Viewing the system as a Markov chain, this corresponds to the fact that the only two recurrent states are **0** and **1** and all the other states are transient[5]. □

Note that the proposition above only gives *sufficient* conditions to solve the problem; moreover, it does not give any idea on the time needed to converge to a fixed point. This means in particular that the function WEST(n) may scale exponentially with n, which is not what is expected for an efficient solution to the problem.

Proposition 7. *Consider the probabilistic ECA ϕ defined with the relationship: $\phi(x,y,z) = (\overline{y} + \overline{z})/2$; this rule is such that: $WEST(n) = \Theta(n^2)$.*

Proof (Sketch). By construction, the rule is the composition of two (commutative) operations: (a) an α-asynchronous left shift with $\alpha = 1/2$ and (b) a global inversion ($x \to \overline{x}$). In the α-asynchronous updating each cell independently applies the local rule with probability α and keeps its state with probability $1 - \alpha$. This rule was analysed in a previous work [**?**] and it was shown that its WECT scales quadratically with n.

It can be verified that the global inversion preserves the dynamics of the asynchronous shift. Indeed, the effect of ϕ is simply to shift the interfaces between regions of 0s and regions of 1s. To formalise this, one could use a coupling between the original process and the asynchronous shift. □

[5] The definitions of recurrent and transient can be found in the introductions to Markov chain theory. Informally, a recurrent state corresponds to a state who is returned to an infinite number of times with probability 1 and a transient state is a state which is not recurrent: it will then be "leaved" definitively with probability 1.

The same construction can be applied to the rules proposed by Fukś and Schüle to solve the density classification problem with stochastic rules. These rules were also analysed and were shown to have a quadratic scaling of their convergence time [**?**].

It is however interesting to note that the Traffic-Majority rule [**?**], whose convergence is conjectured to be linear with the ring size, *does not* obey the invariance by global inversion. Intuitively, the reason of this non-symmetry is that the Traffic rule (i.e., ECA 184) treats the 0s and 1s differently and that a global inversion also reverses the direction in which each state is "translated". It is an open question to find a rule with a synchronisation time that has a linear scaling.

7 Questions

We studied the global synchronisation problem and listed a few simple properties the potential solutions. By formulating the problem as a SAT problem, we could perform a first systematic exploration of the existence of perfect solutions and give a different point of view than the techniques that have been used so far (see e.g. Ref. [**?**]).

We also noted the existence of a huge gap between deterministic and stochastic systems: the use of randomness allows one to easily obtain a "perfect" solution in the sense that any configuration will be almost surely synchronised in finite time. The precise estimation of the average time of convergence is a delicate operation in all generality but insights could be given for some precise rules.

We end this note with a list of questions and indications:

Question 1. Does there a exist a deterministic rule which synchronises all sizes?

As many researchers do, we believe that the answer to this question is negative.

Question 2. If the answer is no, given a neighbourhood of size k, what is the maximum ring size that can be synchronised? Is this function computable?

We have absolutely no hint on how to answer this question.

Question 3. Given a neighbourhood of size k and maximum synchronisation time τ, what is a good algorithm to find all the rules with a height less or equal to τ? What is the complexity of this problem?

The work presented here with the use of SAT solvers can largely be improved. This is only a first attempt to use such techniques. Research could be continued by looking for other symmetries of the problem or other ways to add the constraints expressed in Prop. **??**. Naturally, other paths have also to be searched.

Question 4. Is there a stochastic solution the global synchronisation problem whose WEST scales linearly with the ring size?

At the moment, we do not see how such a rule could be constructed.

Question 5. To which general context is it worth to generalise the global synchronisation problem?

One may think of higher dimensions, more states, non-homogeneous rules[6], Boolean networks, etc.

Acknowledgements

The author is grateful to Irène Marcovici, Jean Mairesse and Sukanta Das for stimulating discussions on the topic and to the anonymous reviewers for their precious remarks and suggestions.

[6] Readers may test ECA 3 with two boundary cells that have a state 0.